Assessment Report

on

“ Classify News Articles by Category”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Dhruv Goel

Roll Number : 202401100300100

Section: B

Under the supervision of

“SHIVANSH PRASAD”

KIET Group of Institutions, Ghaziabad

22,April, 2025

---

1. Introduction

The objective of this project is to classify news articles into predefined categories such as "tech," "business," and "sports." The dataset contains basic metadata about each article, such as word count, whether keywords are present, and estimated read time. This classification problem falls under supervised machine learning and specifically involves multi-class classification. Such systems are useful in news aggregation apps, content recommendation, and digital archiving.

---

2. Problem Statement

In the age of digital media, categorizing news articles efficiently is essential for personalized content delivery, content organization, and search optimization. The aim of this project is to develop a machine learning model that can accurately classify news articles into predefined categories such as *tech*, *business*, and *sports* based on simple metadata like word count, presence of keywords, and estimated read time. Automating this classification process can significantly reduce manual labor and improve the scalability of content management systems.

---

3. Objectives

- **To develop a machine learning model** that classifies news articles into categories such as *tech*, *business*, and *sports* based on structured metadata.

- **To preprocess and encode** categorical data for compatibility with machine learning algorithms.

- **To implement and train** a supervised classification algorithm (Random Forest) on the given dataset.

- To evaluate model performance using standard metrics like accuracy, precision, and recall.
- **To visualize the classification results** using a confusion matrix heatmap for clear interpretation of model effectiveness.

---

4. Methodology

- **Dataset Loading:** Loaded news_articles.csv, which contains 100 entries and 4 columns: word_count, has_keywords, read_time, and category
- **Data Preprocessing:** Preprocessing: Encoded the categorical labels using LabelEncoder.
- **Model Building**: Split the data into training and testing sets (80:20) and trained a RandomForestClassifier.
- **Model Evaluation**: Evaluated the model using accuracy, precision, and recall. Generated a confusion matrix and visualized it with a heatmap.

---

5. Data Preprocessing

The dataset is cleaned and prepared as follows:

- Loaded the dataset using pandas and confirmed no missing values were present.
- Encoded categorical labels using LabelEncoder to convert text labels into numerical format.
- Selected key features: word_count, has_keywords, and read_time for model training.
- The dataset is split into 80% training and 20% testing.

---

6. Model Implementation

A Random Forest Classifier was chosen for implementing the news article classification task due to its robustness and ability to handle multi-class problems effectively. The model was trained on the selected features—word_count, has_keywords, and read_time—from the preprocessed training data. Using scikit-learn's implementation, the classifier was trained with default parameters. Once trained, the model was used to predict the categories of the test data. The performance was then evaluated using standard metrics such as accuracy, precision, and recall, along with a confusion matrix for visual insight into the classification results.

---

7. Evaluation Metrics

The following metrics are used to evaluate the model:

- **Accuracy** measures how often the model correctly classifies news articles overall.

- **Precision**: indicates how many articles labeled by the model as a category were actually correct.

- **Recall**: shows how well the model captures all relevant articles in each category.

- **F1 Score** balances precision and recall into a single metric, especially useful when the dataset has uneven class distribution.

- **Confusion Matrix**: visually maps true versus predicted categories to highlight classification errors.

---

8. Results and Analysis

- The model achieved an accuracy, precision, recall, and F1 score of **0.40**, indicating moderate performance on the news article classification task.

- The confusion matrix revealed that several categories were frequently misclassified, suggesting overlapping feature patterns.

- Despite basic preprocessing, the limited dataset size and simple feature selection likely constrained the model's classification power.

---

9. Conclusion

In conclusion, the project successfully demonstrated the application of supervised machine learning for news article classification. Using a RandomForestClassifier with basic features like word count, presence of keywords, and read time, the model achieved an accuracy, precision, recall, and F1 score of 0.40. While the results indicate room for improvement, they highlight important insights about feature relevance and data limitations. Expanding the dataset, engineering more sophisticated features, and experimenting with advanced models could significantly enhance future performance. Overall, this project provided valuable hands-on experience in data preprocessing, model training, and evaluation within a real-world context.

---

10. References

- Dataset: Provided as news_articles.csv

- Libraries: scikit-learn, pandas, matplotlib, seaborn

- Image: Confusion matrix generated using seaborn heatmap

- Developed using Python 3 in Colab Notebook

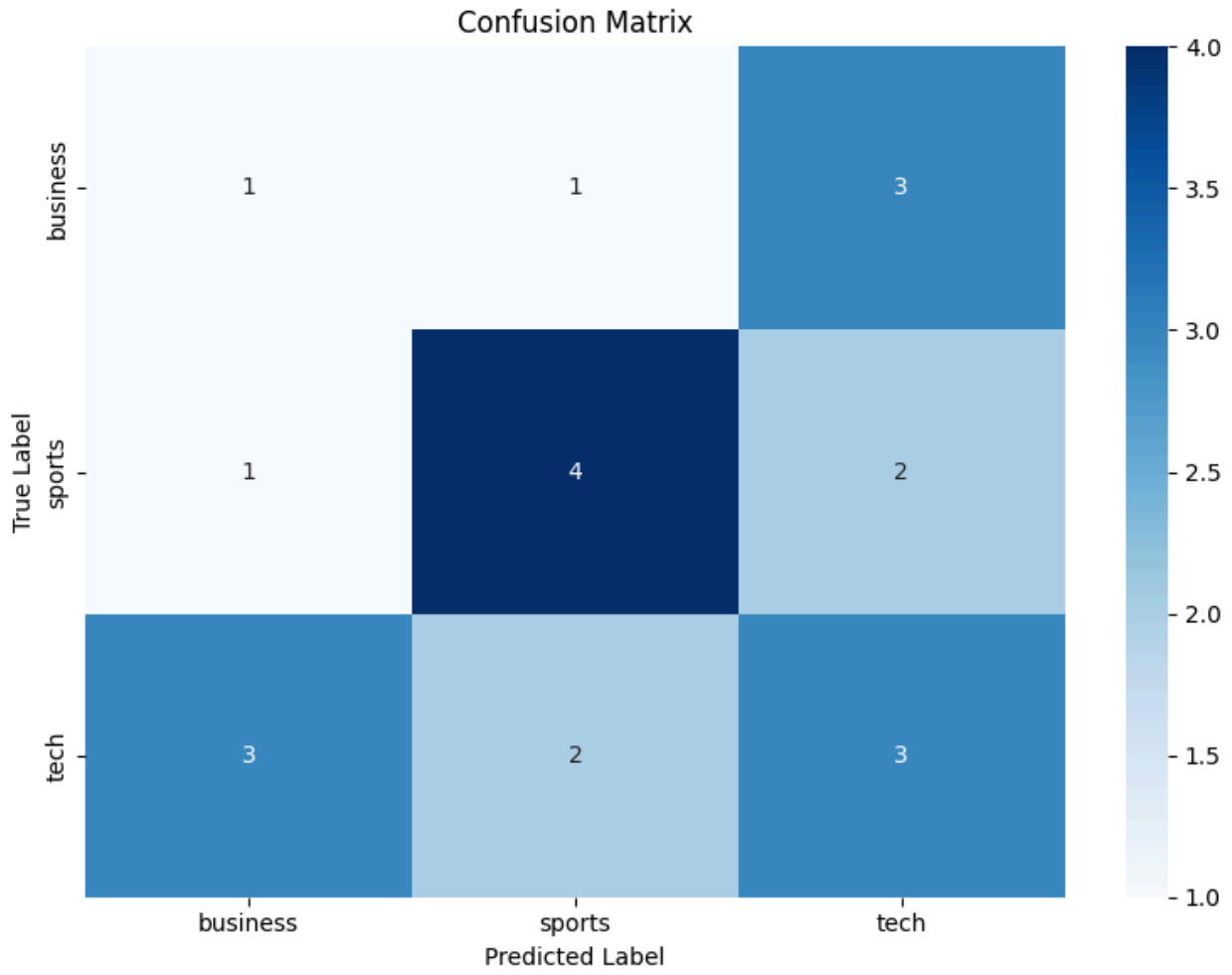
CODE:

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score

import seaborn as sns

import matplotlib.pyplot as plt


# Load Dataset

df = pd.read_csv("news_articles.csv")

X = df[['word_count', 'has_keywords', 'read_time']]

y = df['category']


# Label Encoding

le = LabelEncoder()

y_encoded = le.fit_transform(y)


# Train/Test Split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y_encoded, test_size=0.2, random_state=42)


# Model Training

clf = RandomForestClassifier(random_state=42)

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)


# Evaluation

acc = accuracy_score(y_test, y_pred)

prec = precision_score(y_test, y_pred, average='weighted', zero_division=0)

rec = recall_score(y_test, y_pred, average='weighted')


# Confusion Matrix

cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(8, 6))

sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=le.classes_, yticklabels=le.classes_)

plt.title("Confusion Matrix")

plt.xlabel("Predicted Label")

plt.ylabel("True Label")

plt.tight_layout()
```

plt.show()

IMAGES:



```
#Import Libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
import seaborn as sns
import matplotlib.pyplot as plt
```

```
[3] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[5] # Load Dataset
    df = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/news_articles.csv")  # Make sure the file is in the same directory
```

```
[6] #Inspect Data
    print(df.info())
    print(df.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   word_count     100 non-null    int64 
 1   has_keywords   100 non-null    int64 
 2   read_time      100 non-null    int64 
 3   category       100 non-null    object
dtypes: int64(3), object(1)
memory usage: 3.3+ KB
None
   word_count  has_keywords  read_time  category
0         142             0          3      tech
1        1043             0          6  business
2         442             1         12    sports
3        1449             1         13      tech
4        1937             1         10      tech
```

[7]
```
#Features and Labels
X = df[['word_count', 'has_keywords', 'read_time']]
y = df['category']
```

[8]
```
#Encode Labels
le = LabelEncoder()
y_encoded = le.fit_transform(y)
```

```
[9] #Split Data
    X_train, X_test, y_train, y_test = train_test_split(X, y_encoded, test_size=0.2, random_state=42)

[10] #Train Classifier
     clf = RandomForestClassifier(random_state=42)
     clf.fit(X_train, y_train)
```

RandomForestClassifier

RandomForestClassifier(random_state=42)

```
[11] #Make Predictions
     y_pred = clf.predict(X_test)

[12] #Evaluation Metrics
     acc = accuracy_score(y_test, y_pred)
     prec = precision_score(y_test, y_pred, average='weighted', zero_division=0)
     rec = recall_score(y_test, y_pred, average='weighted')

[13] print("Accuracy:", acc)
     print("Precision:", prec)
     print("Recall:", rec)
```

```
Accuracy: 0.4
Precision: 0.4
Recall: 0.4
```

```
#Confusion Matrix Heatmap
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=le.classes_, yticklabels=le.classes_)
plt.title("Confusion Matrix")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.tight_layout()
plt.show()
```